OLCF6 Benchmark

FORGE: Pre-Training Open Foundation Models for Science

This document provides description about FORGE as a benchmark application, which aims to stress test system capability on low precision compute, communication bandwidth, as well as overall energy usage and efficiency. Accompanied with the benchmark is a suite of reference code that runs at scale, the dataset, and varying model size and downstream evaluation tasks.

Overview

FORGE is an application that pre-trains large language models (LLMs) on 200M scientific papers. It's based on Generative Pre-trained Transformer (GPT) architecture (the same as GPT-NeoX), and runs distributed with data, tensor, and pipeline parallelisms. The resulted foundation models demonstrate good zero-shot performance, and are finetuned for scientific downstream tasks, such as domain-subject classification, etc.

FORGE comes with 3 different model sizes: 1.44B (S), 13B (M) and 22.4B (L). As a rule of thumb, the memory footprint for training FORGE models is about 12 times of the model parameter size. FORGE-S requires about 20GB memory with the vanilla data parallelism, while FORGE-L requires certain level of model parallelisms during training. The larger the model, the more demanding to the system's compute, communication, and energy efficiency. FORGE models utilize GPT-NeoX architecture, which largely follows GPT-3 with some small deviations. These deviations include using rotary positional embeddings instead of absolute positional embeddings, parallel computation of the attention and feed-forward layers rather than executing sequentially, and using full dense layers. Specifically, the detail architectures of FORGE models are listed as follows,

Model	# parameters	# layer	# heads	hidden size
FORGE-S	1.44 B	24	24	2064
FORGE-M	13 B	40	40	5120
FORGE-L	22.4 B	48	48	6144

System Stressing Points

- Low precision compute: FORGE is computationally intensive ($\boldsymbol{Total\_compute\_needed} = 6 \times \#parameters \times \#tokens$), and the computation is in either float16 or bfloat16. Some key kernels such as attention and softmax are optimized by custom CUDA/HIP implementation.
- Communication bandwidth: FORGE is communication bound at scale, especially for large models that can't fit into a single node. The message size is about 3 times of the parameter size, and the main communication patterns are AllReduce, AllGather, and ReduceScatter.
- Energy usage: Since the LLM training requires a huge number of floating-point operations, it is important and a community practice to report the energy usage and the corresponding carbon footprint.

Metrics for evaluation

For each system stressing point, we introduce a measurable metric,

- Computation performance (TFLOPS): Given the floating-point operations per iteration, FORGE will provide the estimation for the training performance in TFLOPS based on the iteration time.

- Scaling efficiency (%): This is defined as the ratio of the measured computation performance (TFLOPS) at scale over the linear extrapolation of the computation performance at the minimum scale. The minimum scale is the minimum number of devices that can fit in the model, and it depends on the model size, e.g., for FORGE-S, the minimum scale is 1 GCD while for FORGE-L it's 1 node on Frontier. As a rule of the thumb, the memory required to fit in a FORGE model is about 12 times of the model parameter size.
- Energy efficiency (TFLOPS/Watt): This is defined as the ratio of the measured computation performance (TFLOPS) during the end-to-end training over the measured power usage (not provided by FORGE, but can be measured by system tools, e.g, nvidia-smi/rocm-smi).

The low-precision training may result into numerical instability. To ensure the training is valid, it is also important to add metrics for the evaluation of the end-to-end results,

- Final loss value: FORGE will log the loss values during the training.
- Time to solution (for a given loss value): This is defined as the time taken to reach a given loss value that we consider as the converged end-to-end training. For FORGE-L, we suggest the final training loss should be smaller than 1.67.

The Figure of Merit (FOM) is defined as the time-to-solution (S1) training FORGE-L. To assist the assessment of the FOM, it is helpful to report additional metrics as defined above (i.e., scaling efficiency and computation performance) during the evaluation of the FOM and FORGE models of other sizes.

Run Rules

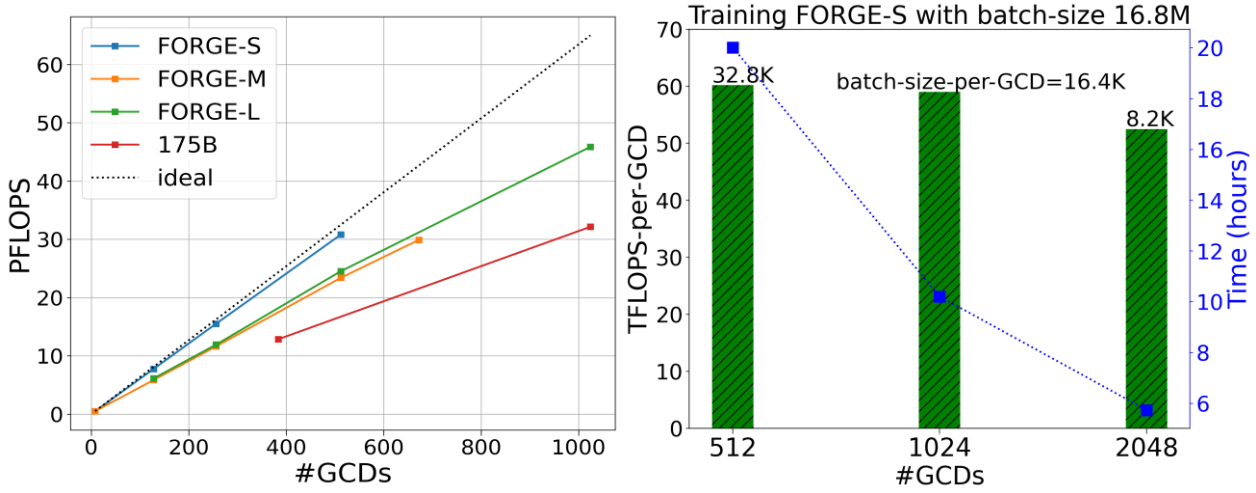
The allowed code modification, including ported and optimized, should follow general OLCF-6 benchmarks run rules. Additionally, to ensure the same computational workload as the baseline, the training parameters are constrained as follow,

- Use Hugging Face byte-pair encoding tokenizer with provided vocabulary file.
- Use Adam or LAMB optimizer.
- Use the same model architectures as FORGE-L.
- Train on all 257B tokens.

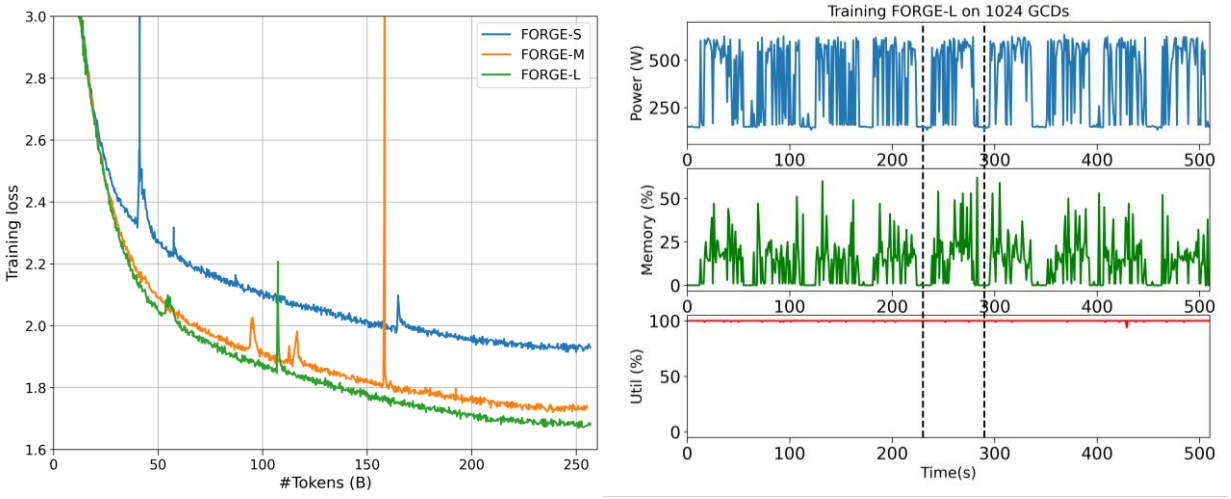
The learning rate and batch size can be adjusted depending on the training scale.

Appendix: Measurements on Frontier

In the following, we provide our initial measurements of FORGE on Frontier, including the computation performance, scaling, energy efficiency, loss value, and time-to-solution.



The scaling of both the computation performance (TFLOPS per GCD and aggregated PFLOPS) and time-to-solution (hours) for FORGE models on Frontier. FORGE-M is 13B model.



The loss values, and power, memory, and GPU utilization for pre-training FORGE models on Frontier. FORGE-M is 13B model.